Enhancement of Ranking and Query Optimizer in Internet Search Engine

Dr. Moaid A. Fadhil
Informatics Institution for postgraduate studies
University of Technology
Baghdad, Iraq
moaid_af@yahoo.com

Abstraction

The World Wide Web (WWW) allows people to share information globally. The amount of information grows without bound. In order to extract information that we are interested in, we need a tool to search the Web. The tool is called a search engine. This research aims to build a ranking system to retrieve the most relevant Web pages with the queries that are presented by the users; the ranking system depends on Web pages content and their hyperlinks' structure. To retrieve the optimal results, the modified ranking system consists of two main subsystems. The first subsystem is Term-based Ranking subsystem that assigns a rank score to the Web pages depending on their contents by implementing TF-IDF Algorithm and Vector Spreading Activation Algorithm. And the second subsystem is Link-based Ranking subsystem that assigns a rank score to the Web pages depending on their hyperlinks by implementing an improved PageRank algorithm. Enhanced Indexing Process is considered as a main tool that supports in retrieving best results.

Keywords: Indexing System, Ranking Algorithm, PageRank, Hyperlink structure, Term frequency, Relevancy.

1. Introduction

Search Engines, which combine such disciplines as database technology, distributed computing and storage, statistical linguistics and graph algorithms, are presented with queries of users. Users expect their queries to be instantaneously answered with ranked lists of the most relevant Uniform Resource Locators (URLs) available online for each query. In order to meet these demands, Web Search Engines (WSEs) must collect and index billions of resources, and develop highly efficient retrieval and ranking algorithms that are capable of effectively answering queries in almost a blink of the eye [1].

Largest WSEs index thousands of millions of multilingual web pages containing millions of distinct terms. Due to the peculiarities and the huge size of the Web repository, traditional Information Retrieval (IR) systems developed for searching smaller collections of structured or unstructured documents appear inappropriate for granting retrieval effectiveness on the Web. Most components of an IR system must be rethought in order to address the problem of effectively and efficiently searching and retrieving information from the Web. Consider, as an example, the problem of deciding which documents are relevant and which are not with respect to a user query. This hard task is commonly delegated to a ranking algorithm which attempts to establish an ordering among the documents retrieved. Documents appearing at the top of this ordering are the ones considered to be more relevant. The two most accepted metrics to measure ranking effectiveness are: Precision (i.e. number of relevant documents retrieved over the total number of retrieved documents) and *Recall* (i.e. number of relevant documents retrieved over the total number of relevant documents in the

collection). In traditional IR, it can be assumed that the documents in the collection originate from a reputable source and all words found in a document were intended for the researcher.

Ranking can simply be based on statistics performed over word frequencies. The same assumption does not hold on the Web where content is authored by sources of varying quality and words are often added indiscriminately to boost the page's ranking. Moreover, as the size of the indexed collection grows, since users usually only look at the first few tens of results, a very high *precision* has to be preferred even at the expense of the *recall* parameter.

2. Related Algorithms

Ranking is the heart of the search engine. Most search engines use variations of the link-based or term-based method to do ranking. Recall that search engines do not allow access to the text, but only the indices, because it is too expensive in terms of time and space. So, ranking must use indices while not accessing the text. Besides, there are also other difficulties as well. It is difficult to compare two search engines, because of their continuous improvement. The exact nature of each search engine's ranking formula is a closely guarded secret [2, 3].

Relevance is determined by weighing both keyword and website popularity factors. The same query done on different search engines will yield different rankings. Additionally, each search engine indexes information in a different way. This means that the relevancy rankings of each site will be unique [3].

2.1 PageRank Algorithm

The original PageRank algorithm was described by Lawrence Page and Sergey Brin in 1998 [4]. The PageRank of a page *p* (denoted *PR(p)*) is the probability of visiting *p* in a random walk of the entire Web, where the set of states of the random walk is the set of pages, and each random step is of one of two types [5]:

i. Choose a Web page uniformly at random, and jump to it.
ii. From the given state *s*, choose uniformly at random an outgoing link of *s* and follow that link to the destination page.

PageRank is defined like the following: Assume that page A has pages T1, T2, ..., Tn which point to it. And parameter d is a damping factor which can be set between 0 and 1 (often d assigned at 0.85). Also, C(A) is considered as the number of links going out of page A. the PageRank of a page A is presented in equation (1) as follows [6]:

$$\left[PR(A) = (1-d) + d \times \left(\frac{PR(T1)}{C(T1)} + \cdots\cdots + \frac{PR(Tn)}{C(Tn)} \right) \right] \quad (1)$$

So, first of all, it is obvious that PageRank does not rank web sites as a whole, they are determined for each page individually. Further, the PageRank of the page A is recursively defined by the PageRank of those pages which link to page A [4, 7]. By recursion, PageRank computations are repeated several times until the number stop changing much. This can be considered as iteration or recursion stopping condition [8].

The PageRank of pages Ti does not influence the PageRank of page A uniformly. Within the PageRank A the PageRank of the page Ti is always weighted by the number of outbound links C(Ti) on page Ti. This means that the more outbound links a page T has, the less will page a benefit from a link to it on page Ti. The weighted PageRank of page Ti is then adding up. The outcome of this is that an additional inbound link for page A will always increase page A's PageRank.

Finally, the sum of the weighted PageRank of all pages Ti is multiplied by the damping factor d. Thereby, the extending of PageRank benefit for a page by another page linking to its will be reduced [4, 7, 9].

2.2 TF-IDF Algorithm

The TF-IDF algorithm is based on the well-known vector space model, which typically uses the cosine of the angle between the document and the query vectors in a multi-dimensional space as the similarity measure. Vector length normalization can be applied when computing the relevance score, $R_{i,q}$, of page P_i with respect to query q.

The vector-length normalization does not work well for short documents. This is consistent with testing queries, if the average length in the test collection is only 48,00 words (not including stop words and other words removed during the indexing process). It may be the case that vector-length normalization does not work for documents where the size of a segment with a coherent topic is small, e.g., a few sentences [10].

$$R(i,q)=\frac{\sum_{termj\in q}\left(0.5+0.5*\frac{TF|i,j|}{TF\max i}\right)IDFj}{\sqrt{\left(\sum_{termj\in Pi}\left(0.5+0.5*\frac{TF(i,j)}{TF\max i}\right)\cdot(IDFj)\right)}} \quad (2)$$

where:

TF(i,j): the term frequency of Qj in Pi.

TF maxi: the maximum term frequency of a keyword in Pi.

IDFj: $\log\left(\frac{N}{\sum_{i=1}^{N}C(i,j)}\right)$ (3)

2.3 Vector Spreading Activation Algorithm

This algorithm is also proposed by Yuwono and Lee in 1996 [2]. It combines the vector space model and spreading activation model. Each document is first assigned a relevance score using TF-IDF algorithm, and then the score of a document is propagated to the documents it references. More formally, the algorithm assigns a relevance score to page P_i with respect to query q as follows in equation (4):

$$R(i,q)=S(i,q)+\sum_{j=1,\,j\neq i}^{N}\alpha*Li(i,j)*S(i,q) \quad (4)$$

where:

N: the number of WWW pages in the index database.

$R_{(i,q)}$: the relevance score of the i-th page respect to the query q

$S_{(i,q)}$: the TF-IDF score of the i-th page

α : is a constant link weight where (0< α < 1).

The relevance score of a document is a sum of the weights of the query terms that appear in the document, normalized by the Euclidean vector length of the document. The weight of a term is a function of a word's occurrence frequency (also called the term frequency) in the document and the number of documents containing the word in the collection (i.e., the inverse document frequency). This weighting function gives higher weights to terms which occur frequently in a small set of the documents. The full vector space model is very expensive to implement, because the normalization factor is very expensive to compute. In TF-IDF algorithm, the normalization factor is not used. That is the relevance score is computed by:

$$R(i,q)=\sum_{termj\in q}\left(\left(0.5+0.5*\frac{TF(i.j)}{TF\max i}\right)\cdot IDFj\right) \quad (5)$$

Through an experiment the value of α is identified in 0.2. The using of recall/precision curves of Vector Spreading Activation on a specific testing queries with α parameter value of 0.0 (as TF-IDF without the spreading activation effect), 0.1, 0.2, 0.3, 0.4 and 0.5 proves that the best retrieval performance is achieved with α equals 0.2 [10].

3. Design and Implementation of Ranking system

Ranking System Overview

The ranking system is consists of two subsystems the first part is link-based ranking and the second part is Term-based ranking subsystem as following:

- **Link-based Ranking**

In This section the link-based ranker will be presented. This subsystem is working in off-line time; it depends on the hyperlink structure of the web page. In this research, several modifications are proposed to

achieve a high quality ranking score by implementing the improved PageRank algorithm on the web pages, These are:

1. **Additional Factors Influencing PageRank.**

The original PageRank calculations are based on the number of inbound links and outbound links only. But by adding the additional factor to the original PageRank equation (1) the improved PageRank will achieve more accurate values. Therefore, the following modification of the PageRank algorithm is assumed:

$$PR(A)=(1-d)+d*\left[\left(\frac{PR(T1)}{C(T1)}\right)*L(T1,A)+\ldots+\left(\frac{PR(Tn)}{C(Tn)}\right)*L(Tn,A)\right] \quad (6)$$

where, L(Ti,A) represents the evaluation of a (inbound or outbound) link which points from page Ti to page A. L(Ti,A) consist of several factors, each of which has to be determined only once. These factors are extracted during the crawler running phase.

1. Visibility of a link which represented by X in equation (8).
2. Position of a link within a web page which is represented by Y in equation (8).
3. Distance between the web pages which is represented by Kd in equation (7).

To implement the evaluation of the linking page into PageRank, the evaluation factor of the modified algorithm must consist of several single factors. For a link that points from page Ti to page A, it can be computed as follows:

$$L(Ti,A)=K(Ti,A)*Kd(Ti) \quad (7)$$

where Kd(Ti) represents the distance between web pages, and K(Ti,A) is the weighting of a single link within a page by its visibility or position, it can be computed as follows:

$$K(Ti,A)=\left(\frac{X(Ti,A)*Y(Ti,A)}{Z(Ti)}\right) \quad (8)$$

where X(Ti,A) represents the position of a link within a document. X(Ti,A) equals 1 if the link is on the lower one third of the page, 2 if the link is on the middle one third of the page, and 3 if the link is on the upper one third of the page. Y(Ti,A) represents the visibility of a link. Y(Ti,A) equals 1 if a link is not particularly emphasized, and 3 if the link is, for instance, bold or italic. Z(Ti) is the summation of the multiplicative correlation between X and Y, it can be given as in equation (9):

$$Z(Ti)=\sum_{K\neq i}(X(Ti,Tk)*Y(Ti,Tk)) \quad (9)$$

Those factors reflect the probability for the random surfer clicking on a link on a specific web page. In the original PageRank algorithm, this probability is given by the term (1/C(Ti)), whereby the probability is equal for each link on one page.

2. **Using most recent PageRank value of each web page.**

The original PageRank equation (1) is implemented in several iterations to be as most as converge. So, in the first iteration all pages rank value is equal to one as an initial value, then in the second iteration pages' rank value at the first iteration is used as an input to the next iteration and so on. Instead of using $PR(t)_{i-1}$ (of *i-1*th iteration) as input to PageRank equation in the *i*th iteration, the most recent PageRank can be used as it is clear in this equation:

$$PR(Pi)=(1-d)+d*\left[\sum_{j<i}\frac{PR(Pj)^{l+1}}{C(Pj)}+\sum_{j\geq i}\frac{PR(Pj)^{l}}{C(Pj)}\right] \quad (10)$$

By implementing these modifications on the original PageRank equation, the proposed PageRank equation will be as follows:

$$PR(Pi)=(1-d)+d*\left[\sum_{j<i}\left(\frac{PR(Pj)^{l+1}}{C(Pj)}*L(Pj,Pi)\right)+\sum_{j\geq i}\left(\frac{PR(Pj)^{l}}{C(Pj)}*L(Pj,Pi)\right)\right] \quad 11$$

where *l* is the iteration number.

Link-Based Ranker subsystem is programmed by ASP implemented in Microsoft Visual InterDev 6.0 environment. It interacts with Links database in Microsoft SQL Server 7.0 by VB Script code which contains SQL query statements. Finally, the

Link-Based Ranker algorithm is as follows:

Link-Based Ranking Algorithm
Input: webpage, and hyperlink
Output: linke-based rank values
[1] while no_error
[2] for all Pi in web retrieve_factors(Pi)
[3] for all Pj points to Pi
[4]
$$PR(Pi) = (1-d) + d * \left[\sum_{j<i} \left(\frac{PR(Pj)l+1}{C(Pj)} * L(Pj,Pi) \right) + \sum_{j\geq i} \left(\frac{PR(Pj)l}{C(Pj)} * L(Pj,Pi) \right) \right]$$
[5] err=abs[PR(Pi)l-PR(Pi)l+1]
[6] if err>0.000001 then
[7] no_error=fales
[8] else no_error =true
[9] end[while]

¬ Term-based Ranker

Term-Based Ranking is done primarily to provide highly relevant pages to the user at the first place and the pages that are not so relevant will be not retrieved to the user unless search option includes this information. Proposal Term-Based Ranking algorithm ranks the retrieved web pages to determine the relevancy of these pages with the query terms. Web pages are ranked based on multiple factors retrieved by the searcher as mentioned. The efficiency of the Term-Based Ranking process can be improved by using two algorithms to compute page's ranking, TF-IDF and Vector Spreading Activation algorithms. In this project, TF-IDF is enhanced by supplementing it with the attribute for each word occurrence. Then the obtained number from the enhanced TF-IDF is considered as a factor in Vector Spreading Activation equation (4).

Important factors are retrieved from the Search Engine Indices by the searcher. One of them is page_id that for each page_id the relevancy is calculated depending on the occurrence of query terms by the improved TF-IDF relevancy equation (12):

$$S(i,q) = \sum_{termj \in q} \left(\left(0.5 + 0.5 \cdot \frac{TF(i.j)}{TF \max i} \right) \cdot IDFj \cdot attributej \right) \quad (12)$$

Equation(1) is improved by adding the term attribute that represents the summation of the following factors:

1. The first factor that is used in computing the attribute is the word importance multiplied by the related word position:
   F1= importance * norpos
   And
   norpos = 1 – (position / nows)
   the absolute number that is used in representing a word position is a critical number because it depends greatly on the number of word in the entire page. For example, suppose the there are two pages , the first one has 100 word as a content and the second has 300 words as a content and in both of them the word "rank" occurs at position 90, it will be clear that the position in the first page is approximately at the end of the page, its relative position is:
   norpos = 1 – (90 / 100) = 0.1
   but the second page the position 90 is approximately at the end of the first one third of the page. Its relative position is:
   norpos = 1 – (90 / 300) = 0.7
   From these two results it is obvious that 0.7 > 0.1 that means the accurate word positioning depends on the page length (in words). Also, the word position is affected by its importance as presented in table 5. So, in the first page, if the word "rank" occurs as a hypertext (link text), its importance will be equal to 6:
   F1 = 6 * 0.1= 0.6
   And in the second page, if "rank" occurs as a normal text its importance will be equal to:
   F1 = 2 * 0.7 = 1.4
2. The second factor represents the color factor: if the word's font color is matched with global page font color the value of F2 will be equal to 1, otherwise it will be equal to 2.
3. The third factor represents the font face factor: if the word's font face is matched with global page font face the value of F3 will be equal to 1, otherwise it will be equal to 2.
4. The fourth factor F4 represents the style of each word, each style has its own value as presented earlier in table 6.
5. and the last one F5 represents the size of each word.

Finally, the attribute will be equal to:

Attribute = $\sum F_{k\ldots\ldots\ldots}$ (13)

Then the term is multiplied to the single iterated term. Then the result that yielded from the improved TF-IDF is considered as an input to the Vector Spreading Activation.

The proposed Term-Based Ranking algorithm will be implemented as follows:

Term-Based Ranking Algorithm
Input: Set of attribute
Output: term-based rank values
[1] for all pages in Retrieved_pages_list
[2] for all terms in Query
[3] for each occurrence of termj in pagei
[4] attributej= term_attribute(factors)
[5] $$S(i,q)=\sum_{termj\in q}\left(\left(0.5+0.5\cdot\frac{TF(i,j)}{TF\max i}\right)\cdot IDFj\cdot attributej\right)$$
[6] $$R(i,q)=S(i,q)+\sum_{j=1,\,j\neq i}^{N}\alpha\cdot Li(i,j)\cdot S(i,q)$$
[8] goto [1]

To determine which Web page is to be displayed first on the result list, the higher number of hits on a Web page determines a higher quality page and will be displayed to the user at the top of the list. This is known as the Rank algorithm. Once the results are gathered and pages are sorted according to the Rank algorithm, the searcher will display them back on the GUI to the user. The URL's of every keyword match are potential results to be displayed to the user.

Implementation Details

On the server side, the process involves creation of a database and its periodic updating done by software called Crawler. The Crawler also called *robot* that crawls the WebPages periodically and indexes these crawled WebPages in the database. In the indexing the keywords are stemmed to remove the inflections redundancy of the word and only the roots of the words are stored in the database. Then all crawled pages are indexed later. As stated earlier, the Indexer processes the web pages through the hyperlinks. In this process it extracts the 'title', 'keywords', and any other related information needed for providing complete search results from the HTML document. Sometimes, the entire content of the HTML document except for the stop words is extracted and indexed in the database.

The idea of indexing the whole page except the stop words is based on the fact that a page dealing with a particular issue will have relevant words throughout its page. Thus indexing all the words in a document increases the probability of getting the relevant WebPages to a query.

The front-end of the search engine is the client side having a graphical user interface as in figure 1, which prompts the user to type in the search query. The interface between the client and server side consists of matching the user query with the entries in the database and retrieving the matched WebPages to the user's machine. One point is worth noting here: before the query words are processed they are stemmed before they are searched for in the database. The database consists of a number of tables that are arranged so as to facilitate faster retrieval of the data. This database is housed in a database server that is called Search Engine Indices, which is connected to the search engine. The typical English search engines will have more than one database server due to the huge number of English web sites. The proposed search engine uses a single database server, because of the small number of web pages. When the user types the query it is taken to the server housing the search engine.

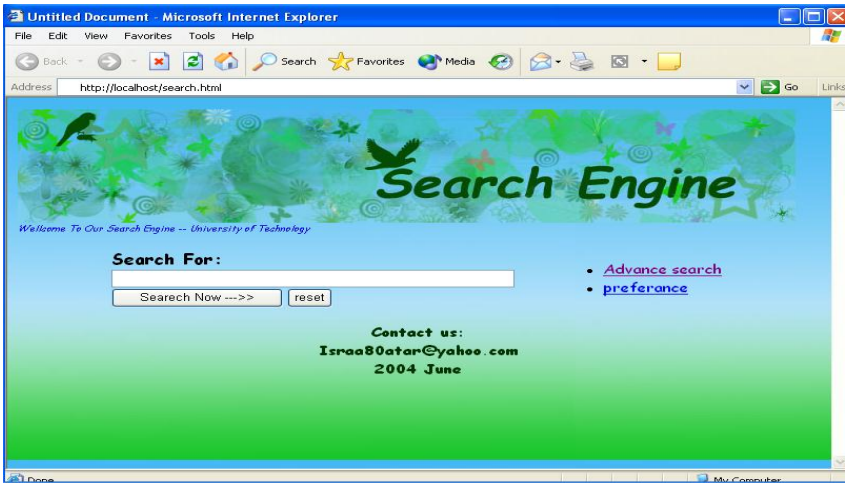

Figure 1 Search Engine User Interface

The proposed search engine validates the query and then translates it into the structured query language (SQL) which is understandable to the database and passes this SQL query to the Search Engine Indices. The Search Engine Indices identify the database entries that match the query given and sends to the proposed search engine server these entries along with other information related to other entries such as the title, the author name, URL and the matching portion from the content of the corresponding entry. The proposed search engine sorts these database entries using a ranking algorithm. The ranking algorithm determines the relevancy of a retrieved webpage to the user query. The retrieved sites are then displayed along with links to these sites and a small portion of text from the matched content. This text gives an idea to the user about the page before the user goes to that particular page.

Major Data Format

It is a database that consists of the following tables:

1. **URL Table:** This table is created during the Crawler running interval.

Table 1 URL Table			
page_id **(unique)**	page_URL **(full address)**	file_type **(.html or .htm)**	crawled **(as a flag)**

This table consists of four fields:

- **page_id**: represents the unique assigned identification number for each individual Web page.
- **page_URL**: represents the full URL of Web page.
- **file_type**: refers to the page's type whether it is .html or .htm.
- **crawled**: works as an acknowledgement flag to distinguish between the crawled and others which have not yet crawled, because of a modification, or new uploading.

2. **Lexicon and Stop_Word Table:** Lexicon Table is used in Inverted Indexing step at the Indexing phase.

Table 2 Lexicon Table	
word _id (unique)	word (stemmed)

This table consists of two fields:

- **word_id**: represents the identification number of each word. These ids are used mapping with page_id in the Inverted Index Table to reduce its size, instead of using the word , its id will be used there.
- **word**: represents the regular words (or stemmed words).

Stop_Word Table contains the noisy words. The word in noisy_word field is used to remove the noisy words from the converted page. The format of the Stop_Word Table is presented in table ٣.

Table 3 Stop_Word Table
noisy_ word

- **noisy_word**: represents the set of words which must be removed from the text file, e.g. the, is, are, on, in, of, ..., etc. .

3. Inverted Index Table : This table is created during the Indexing phase.

- **page_id.**
- **word_id.**
- **position:** is the offset of the word within the web page. The number in this field is related to the length of page.
- **importance:** The importance of a word is determined by specifying the location where the word occurs.

Table ٥ Importance Value Table

Word Location	Word Importance
TITLE tag or title property in META tag.	10
Keywords in META tag.	8
Text used in the link, between <a> and tags.	6
Text in Alt attribute in the <IMG> tag.	4
Normal text.	2

- **style:** This attribute represents (Bold, Italic, Underline) styles each case has its unique identified number. For example, if the style of a specific word is (Italic), the word's style value will be equal to 2. The style value is presented in the table ٦ below:

Table ٦ Style Value Table

Bold	*Italic*	Underline	Style Value
False	False	False	0
False	False	True	1
False	True	False	2
False	True	True	3
True	False	False	4
True	False	True	5
True	True	False	5
True	True	True	6

- **color:** represents the font color for the current word.
- **face:** represents the font face for the current word.
- **size:** represents the font size for the current word.

4. Page_Information Table:

Table ٧ Page_Information Table

page_id (unique)	Page _title	Author _name	Keywo rds	Descry ptor	Modific ation_date	Publis hing_date

- **page_id.**
- **page-title:** the title is extracted either from the <TITLE> tag or from the <META> tag when its attribute **name** equals title then the content of the **content** attribute represents the web page title for example:

  <META name= "title" content= "Web Developer.com guide to building intelligent web site with JavaScript">

Table 4 Inverted Index Table

page_id **(unique)**	word_id **(unique)**	position **(offset)**	Importance	Style	color	face	size

- **author_name:** the name of the author is extracted from the <META> tag when its attribute **name** equals author then the content of the **content** attribute represents the web author name. for example:

  <META name= "author" content= "Nigel Ford">

- **keywords:** the keywords are extracted from the <META> tag when its attribute **name** equals keyword then the content of the **content** attribute represents the most frequently used word which is related to the topic of the web page.
- **descriptor:** the descriptor is extracted from the <META> tag when its attribute **name** equals descriptor then the content of the **content** attribute represents a simple description about the subject of the web page.
- **modification_date:** Modification date refers to the date that the web page is uploaded again to the crawler because of any simple modification or updating happening on this web page. It is extracted from the <META> tag when its attribute **name** equals (update,

modfdate, or any word refers to the modification_date), then the content of the **content** attribute represents the web modification date.

- **publishing_date:** Publishing date refers to the date that this web page is published or uploaded at first time to the crawler. It is extracted from the <META> tag when its attribute **name** equals (pubdate, publishdate, or any word refers to the publishing_date), then the content of the **content** attribute represents the web publishing date.

5. Page_Form_Information Table:

Table ^ Page_Form_Information Table

page-id **(unique)**	number of words **(nows)**	color	face	size

- **page_id.**
- **number of words:** abbreviated to nows; refers to the total number of different words occurred in the page_id specified page.
- **color:** represents the general font color for the current page_id.
- **face:** represents the general font face for the current page_id.
- **size:** represents the general font size for the current page_id.

This database is designed in Microsoft SQL Server 7.0 and manipulated by SQL-T.

4. Evaluation

Evaluating of the search result is to measure how well the returned results meet the user's particular information need. Here two metrics to evaluate the performance of the search engine from two aspects.

Evaluation Metrics And Experiment and Results

To evaluate the proposed search engine, there are two main classes of performance measures, *Effectiveness* and *Efficiency*. Also, from the software viewpoint the Understandability of the User Interface is considered an important issue:

- Effectiveness

Evaluating of the search result is to measure how well the retrieved results meet the user's particular information need. There are two standard measurements Recall and Precision that are used in evaluating the performance of the proposed search engine. Recall and Precision are based on human –relevance judgments and are thus difficult to establish unless such a judgment is readily available. To explain these principles, some examples will be introduced. P is the set of all relevant web pages in existence at a certain point of time. R is the set of all results returned for a search at the same time. C is the set of all relevant results. p, r, and c are defined as the count of the capitalized sets (e.g. p is the amount of elements in P).Recall and Precision could be defined as:

¬ Recall

Recall determines the percentage of relevant documents that were retrieved. The Recall value is between 0 and 1. It defines as:

$$\mathrm{Re}call = \frac{Number \cdot of \cdot relevant \cdot documents \cdot retrieved}{Number \cdot of \cdot relevant \cdot documents} = \frac{c}{p}$$

A high recall means the most of the page that should be returned by a perfect search engine is returned. While in normal or in advanced mode all results that are presented in the Search Engine User Interface are still used. Those pages must have a rank score higher than 20% to be retrieved. The full

Table 9 Searching result, Recall and Precision measures.

Query	Results	Recall	Precision
Web page rank	All first 10 pages are relevant	80%	100%
"ASP Courses"	1,2,3,4,5,6 very Relevant 7,8,9,10 Relevant	90%	100%
Music news	1,2,3,4 Very Relevant	100%	100%
HTML and Java Script	1,2,3,4,5 Very Relevant 6,7,8,9,10 Relevant	70%	100%
Internet not Intranet	1,2 Very Relevant 3,4,5,6,7,8,9,10 Relevant	50%	100%

evaluation of the recall can only be done by doing a user plane review or (by a user judgment).

¬ Precision

Precision is a measure that shows how much of what the user sees is relevant. The resulting value is a real number between 0 and 1. Precision is very important to the proposed search engine given thousands of web pages. This measure is defined as:

$$precision = \frac{Number \cdot of \cdot relevant \cdot documents \cdot retrieved}{Number \cdot of \cdot relevant \cdot documents} = \frac{c}{r}$$

In this research, five different kinds of queries are tested and evaluated by Recall and Precision of the first 10 retrieved results.

The results show that the general Precision of the first 10 retrieved web pages is always 100%, which is reasonably good. From the Precision results it is obvious that the rank values of the retrieved web pages reflect the real relevancy of the existent web pages in the proposed search engine databases.

- Efficiency

Efficiency measures the time that a system requires to find relevant information. The simplest iterative method for solving equation (1), is required to compute at k iterations. But by improving the original PageRank algorithm as presented in equation (4.6) the Gaus-Seiled iteration is used that uses the most recent value wherever possible. In figure 2 the convergence is plotted of two equations (3.2) and (4.6) with damping factor equal to 0.85 (d=0.85).

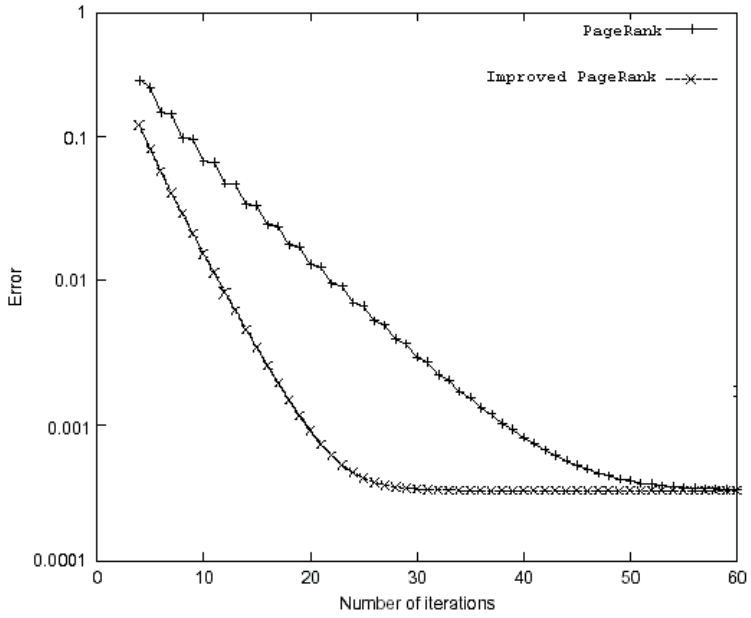

Figure 2 Comparison of the convergence degree between the original and Improved PageRank Algorithm with (d=0.85)

5. Conclusion

1. By implementing the proposed search engine with different kinds of queries, the yielded results prove that the aim of this research is achieved in retrieving the most relevant web pages to the optimized user query.

2. By improving the PageRank algorithm (Link-based Ranker), the number of iterations to reach to the convergence is reduced.

3. The retrieved documents in this research are limited on web pages or web sites only (Files of type *.htm and *.html).

4. References

1.	**Lempel, R.**. ***Link Structure analysis and Query Result caching in Web Search Engines***. ph. D. in Computer Science ,Institute of Technology. Haifa. November 2002 . Available at:
	http://www.webir.org/resources/phd/Lemel-phD-2002.pdf
2.	**Lam, S.** ***The Overview of Web Search engines***. February 9, 2001. Available at:
	http://db.uwaterloo.ca/~tozsu/courses/cs748t/surveys/sunny.pdf
3.	**Dennis O'Connor.** ***Rankings – How are Search Results Listed?*** In Proceeding Illinois Mathematics and Science Academy. September 30, 2003. Available at:
	http://21cif.imsa.edu/inform/totw/tip5/ranking.pdf
4.	***A Survey of Google's PageRank***. Available at:
	http://www.discountdomainsuk.com/articales-archived/7/200/1
5.	**Lempel, R., Moran, S.**. ***Rank-Stability and Rank-Similarity of Link-Based Web Ranking Algorithms in Authority-Connected Graphs.*** Kluwer Academic Publishers. Printed in the Netherlands. stab_kluwer.tex; August 3, 2004. Available at:
	http://www.cs.technion.ac.il/~moran/r/PS/stab-kluwer.pdf
6.	**Craven, P.** ***Google's PageRank Explained and how to make the most of it.*** Available at:
	http://www.webworkshop.com/pagerank.html
7.	**Brin, S., Page, L.** ***The Anatomy of a Large-Scale Hypertextual Web Search Engine***. Computer Science Department, Stanford University, Stanford. 1998. Available at:
	http://www-db.stanford.edu/google.pdf
8.	**Rogers, I.**. ***The Google PageRank Algorithm and How it Works.*** May 16, 2002. Available at:
	http://btoq.csdn.net/ewaves/archive/2004/06/03/30325.aspx
9.	**Desmond J. Higham, Taylor, A.**. ***The Sleekest Link Algorithm.*** University of Strathclyde Mathematics Research Report 20. August 28, 2003.
	http://www.charlesdce.com/pagerank.pdf
10.	**Ming-Yen Thomas Su** ***Item Selection By "Hub-Authority" Profit*** Ranking MASTER OF SCIENCE of Computing Science September 2002. Available at:
	http://www.cs.sfu.ca/~wangk/pub/kdd-kewang.pdf